HackBio

Programming for Life Scientists

Winfred Gatua

18/03/2021

Objectives

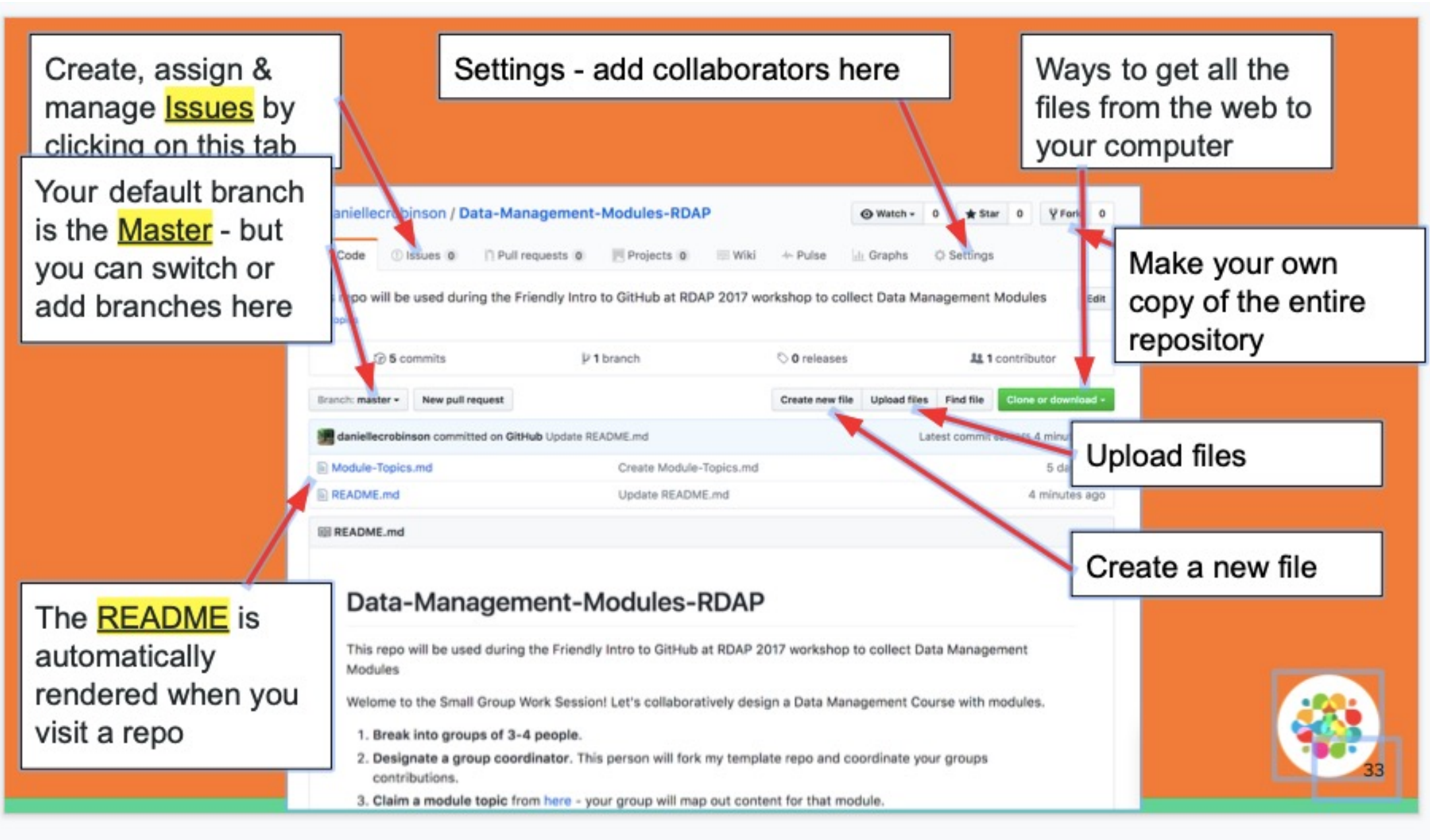
- Git and GitHub
- Linux
- High Performance Computing
- Python

Git and GitHub

- Git – used for version control
- GitHub –host the services

- Practicing Markdown

- This is a **hands on** class and here is the reference material

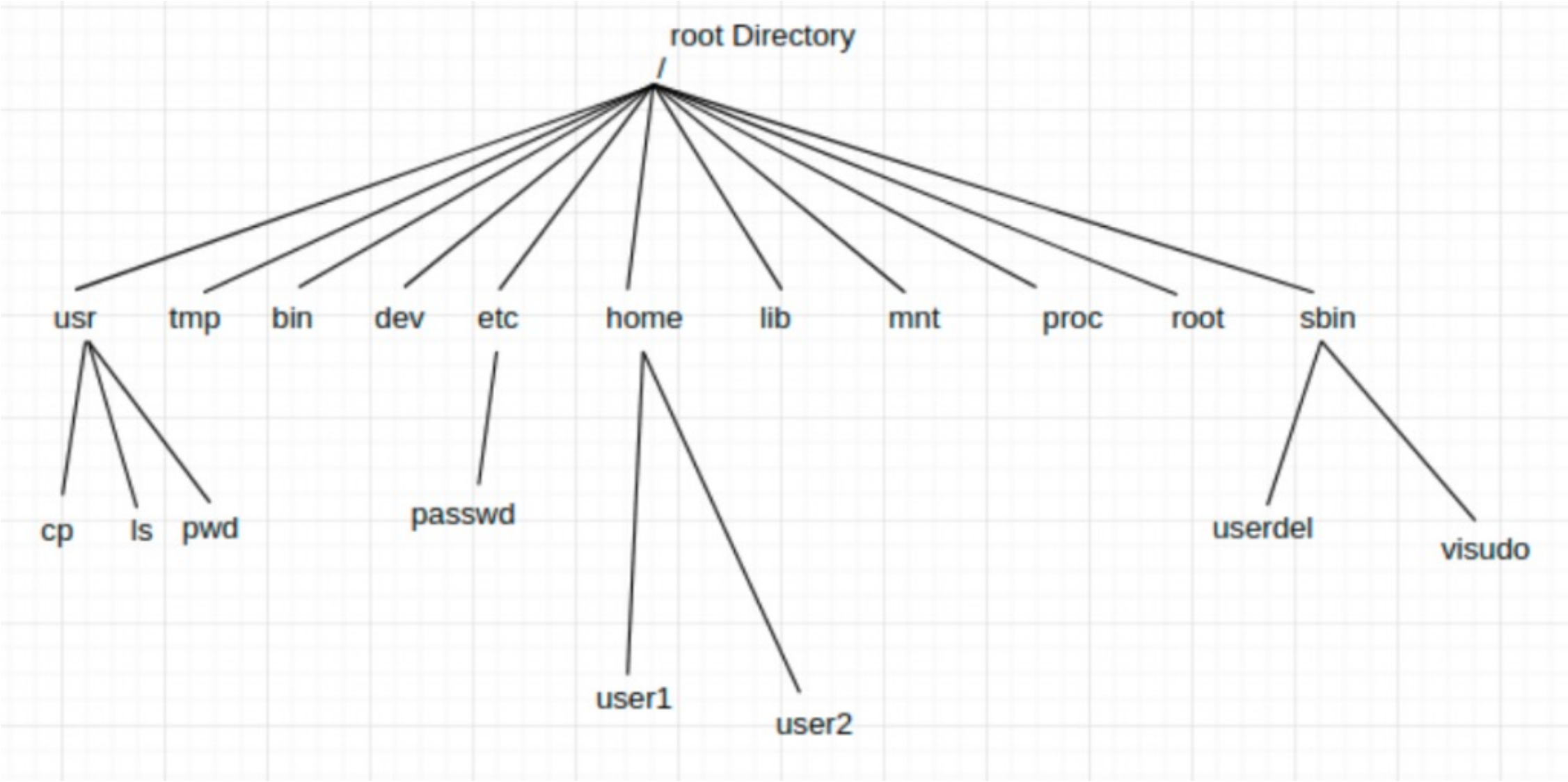

Create, assign & manage Issues by clicking on this tab
Settings - add collaborators here
Ways to get all the files from the web to your computer
Your default branch is the Master - but you can switch or add branches here
Make your own copy of the entire repository
Upload files
Create a new file
The README is automatically rendered when you visit a repo
Data-Management-Modules-RDAP
This repo will be used during the Friendly Intro to GitHub at RDAP 2017 workshop to collect Data Management Modules
Welome to the Small Group Work Session! Let's collaboratively design a Data Management Course with modules.
1. Break into groups of 3-4 people.
2. Designate a group coordinator. This person will fork my template repo and coordinate your groups contributions.
3. Claim a module topic from here - your group will map out content for that module.

Linux

- It refers to a generic term referring to Unix-like GUI based computer operating systems
- Its multiuser, multitasking, multiprocessor
- Has Windows like GUI
- It coexists with other operating systems (OS)
- Runs on multiple platforms
- OS – allows communication between computer hardware and the software

Linux File system

HackBio



Basic Linux Commands

- File handling
- Text processing
- System administration
- File systems
- Advanced commands

How to learn commands

- There are two ways you can approach this:
- Man (Manual) pages
  - Man <command>
  - <Command> help

File Handling Commands

- **Mkdir** – make directories
- Usage: mkdir [OPTION] DIRECTORY…

- **ls**- list directory contents
- Usage: ls [OPTION]… [FILE]..

- **Cd** – change directories
- Usage: cd [DIRECTORY]

Continue

- **Pwd** – print name of current working directory
- Usage: pwd
- **Vim**- Vi improved, a programmers text editor
- Usage: vim [option] [file]
- **#alternatively**
- **Nano** [OPTIONS]

- **NB: Never write any code on Microsoft word instead use text editors**

Continue

- **Cp-** copy files and directories
- **Usage:** cp [OPTION]… SOURCE DEST
- **Mv** – move (rename) files
- Usage: mv [OPTION]… SOURCE DEST
- **Rm** –remove files or directory (**No recycle bin**)
- Usage: rm [OPTION]… FILE
- **Find** – search for files in a directory hierarchy
- Usage: find [OPTION] [PATH] {PATTERN]
- **History**- prints recently used commands
- Usage: history

Text processing

- **Cat** – concatenate files and print on the standard output
- Usage: cat [OPTION] [FILE]
- **Echo**- display a line of text
- Usage: echo [OPTION] [string]
- **Grep** print lines matching a pattern
- Usage: grep [OPTION] PATTERN [FILE]
- **Wc** - print the number of newlines, words, and bytes in a files
- Usage wc [OPTION] [FILE]

System administration

- **Chmod –** change file access permissions
- Usage: chmod [OPTION] [MODE] [FILE]
- **Chown** – change file owner and group
- Usage:chown [OPTION] …OWNER[:[GROUP]] FILE
- **Su**- change user ID or become superuser
- Usage: su [OPTION] [LOGIN]
- **Passwd**- update a user’s authentication tokens(s)
- Usage passwd [OPTION]
- **Who**- show who is logged on
- Usage: who [OPTION]

Advanced commands

- **Reboot-** reboot the system
- Usage: reboot [OPTION]

- **Poweroff**- power off the system
- Usage: Poweroff

Introduction to High Performance computing

Winfred Gatua

18/03/2021

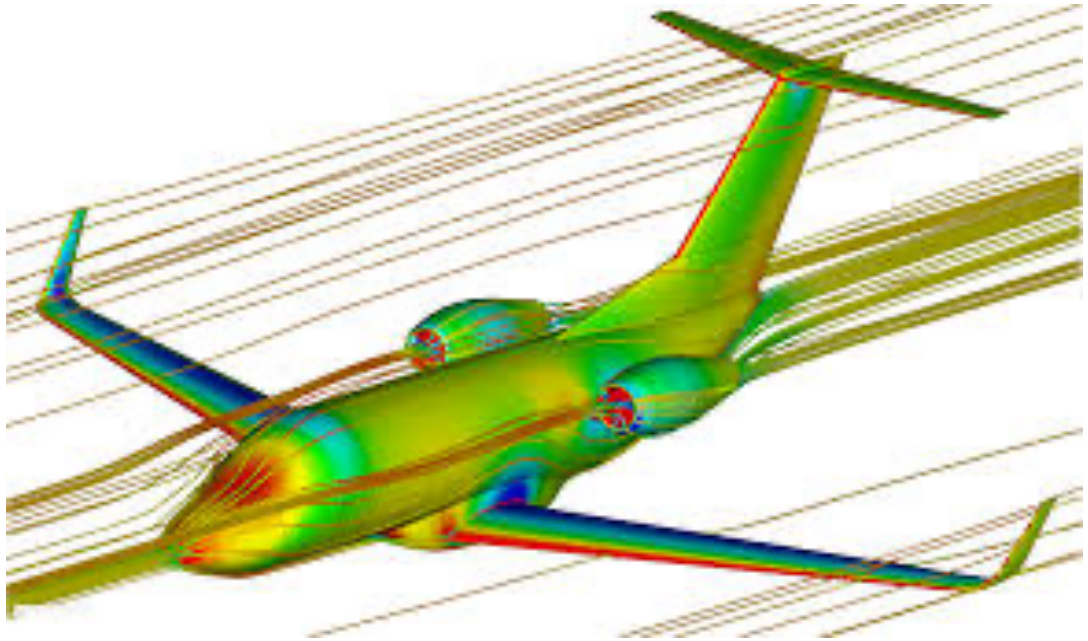
High Performance computing

What is High Performance Computing?

- High Performance Computing (HPC)
  - Networking and Storage
    - Deals with high and highest performance computers, with high speed networks, and powerful disk and tape storage systems
  - Performance improvement
    - Compared to personal computers and small workstations:
    - Factor 100...10.000

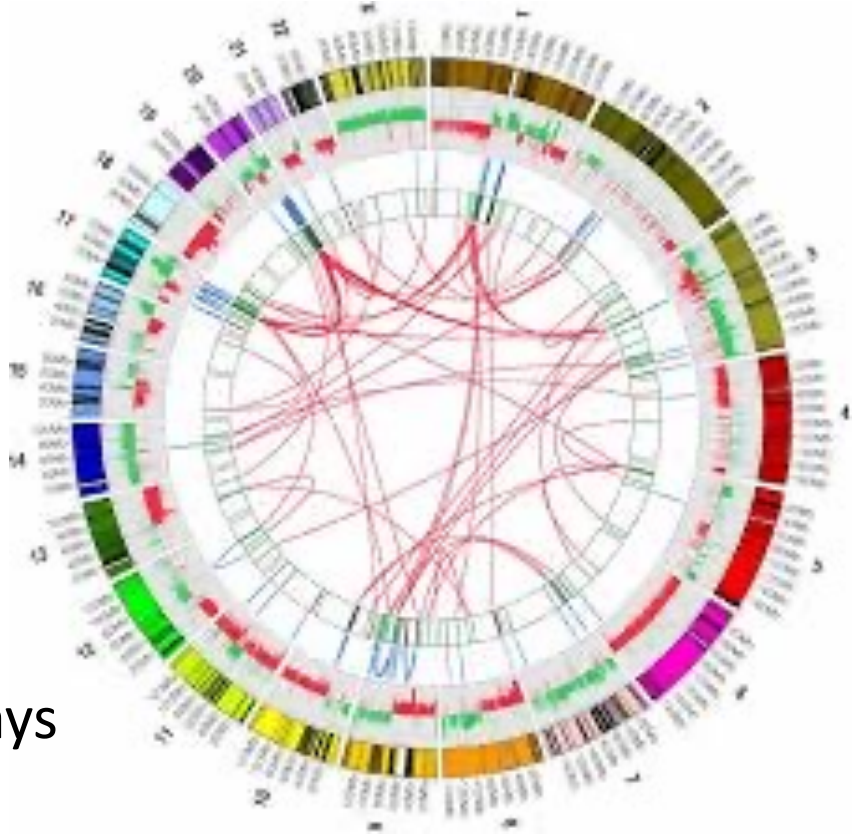
Application Fields

- Numerical calculations and simulations
  - Particle physics
  - Computational fluid dynamics
  - Car crash simulations
  - Weather forecast
  - Commercial database applications
- Non-numerical computations
  - Chess playing, theorem proving
  - Commercial database applications



Continue

- All fields of Bioinformatics
  - Computational genomics
  - Computational proteomics
  - Computational evolutionary biology
- In general
  - Everything that runs between 1 and 10.000 days
  - Everything that uses high volumes of data



Measures

- Mega (10^6) – Giga (10^9) – Tera (10^{12}) – Peta (10^{15}) – Exa (10^{18})
- Computational Performance (Flop/s)
  - Flop/s = floating point operations per second
  - Modern processor: 3 GFlop/s
  - Supercomputer: 1.88 ExaFlop/s (factor 1,000,000,000)
- Network performance (Byte/s)
  - Personal computer: 10/100 MByte/s
  - Supercomputer networks: gigabytes/s

Types of computing problems

Tightly Coupled

1 + 1 = X

X + 2 = Y

Y + 3 = Z

N + Z = W

Embarrassingly Parallel

1 + 1 = X

2 + 2 = Y

3 + 3 = Z

N + N = W

Generic bioinformatics use case

Embarrassingly Parallel

1 + 1 = X

2 + 2 = Y

3 + 3 = Z

N + N = W

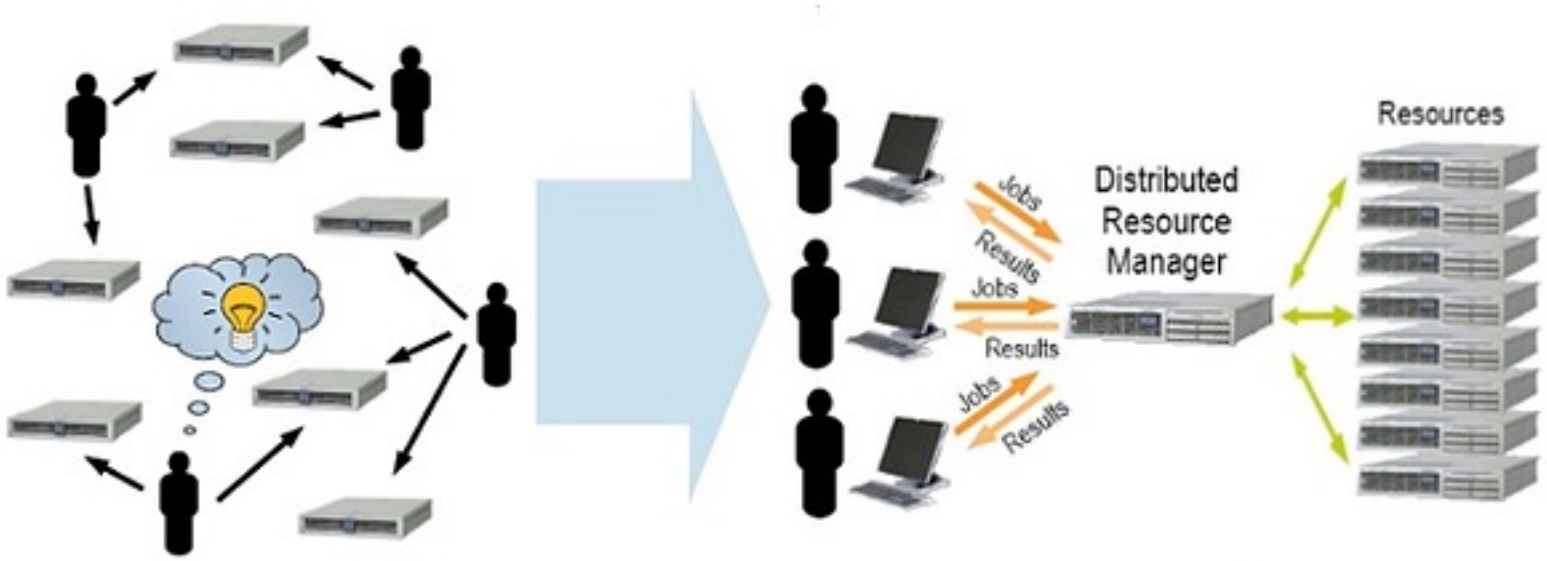
- Scientist has a problem
- She does not want or need to run ONE parallel application stretching across a 100 CPU cluster
- She wants to run one standalone program 100,000 or 1,000,000 times with slightly different input and output values

Embarrassingly parallel tasks
lend themselves well to
“high performance computing”

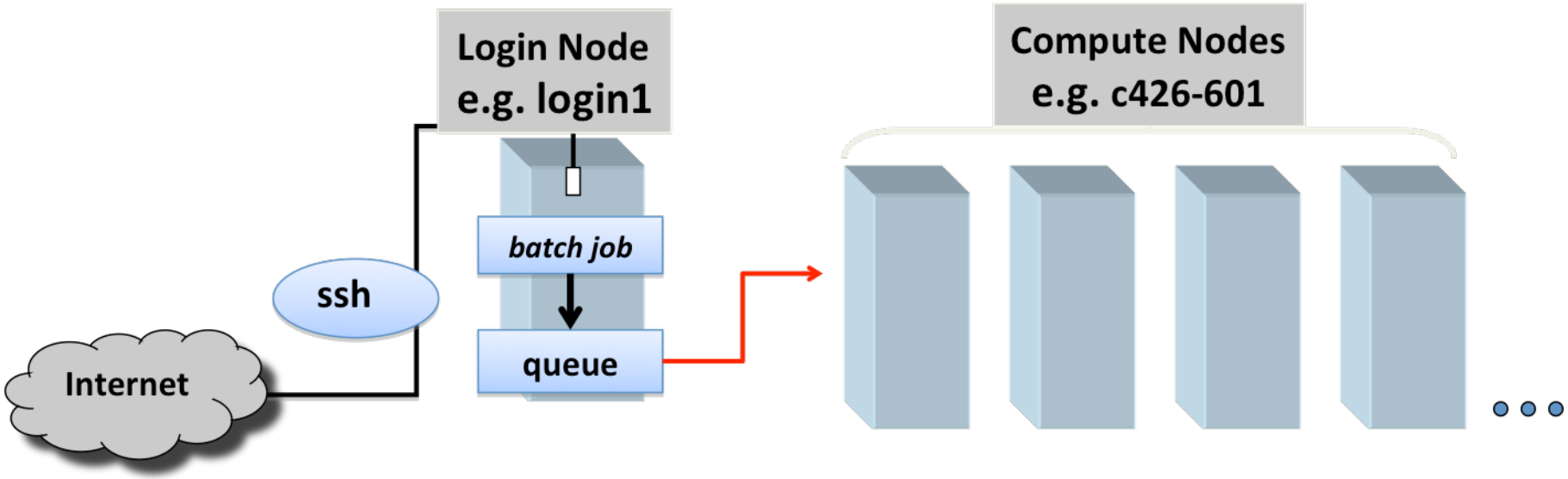
High Performance Linux cluster Job management

Unmanaged

Managed by SLURM



Job Submission with SLURM



SLURM workload manager

- Slurm is an open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters
- Slurm has three key functions:
    1. Allocates exclusive and/or non-exclusive access to resources (compute nodes)
    2. Provides a framework for starting, executing, and monitoring work processes on the set of allocated nodes
    3. Arbitrates contention for resources by managing a queue of pending work

SLURM commands

- **sinfo -** reports the state of partitions and nodes managed by SLURM
- **smap** - reports state information for jobs, partitions, and nodes
- **sbatch** - is used to submit a job script for later execution
- **squeue** - reports the state of jobs or job steps.
- **srun** - is used to submit a job for execution in real time
- **scancel** - is used to stop a job early.

Job Submission with SLURM

- The user “submits” the job to the RMS e.g. issuing

  “**sbatch jobscript.pbs**”

- The user can control the job
  - sbatch: submit
  - sinfo: poll status information
  - scancel: cancel job
- It is the task of the resource management system to start a job on the required resources
- Current system state is taken into account
- Can also submit interactive jobs
  - reserve CPU and memory for use

Example: SLURM Job Description

- Simple job script:

```
#!/bin/bash #
#SBATCH -n 10 # Number of cores
#SBATCH -p longrun
#SBATCH --mem-per-cpu 1024
#SBATCH --mail-user=wgatua@hackbio.org


#Commands to run. Replace example below with your arguments and
variables.


module load python/3.4.3
module load blastall/2.5.6


blastall –I query_seq.fa -d nr_database -p blastp
```

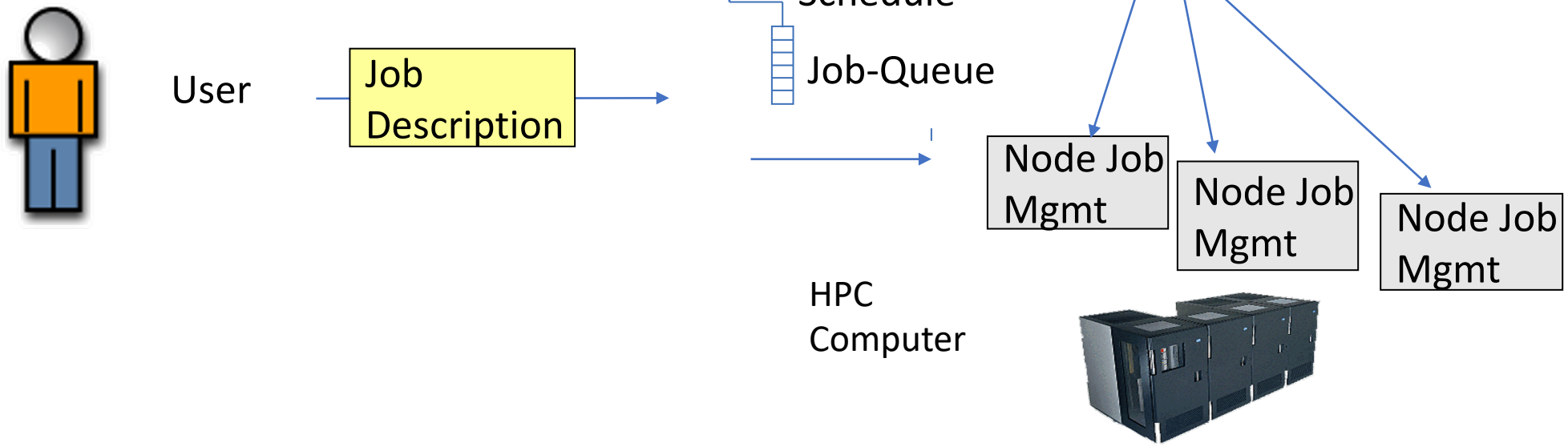
whole job file is a shell script

information for the RMS are comments

the actual job is started in the script

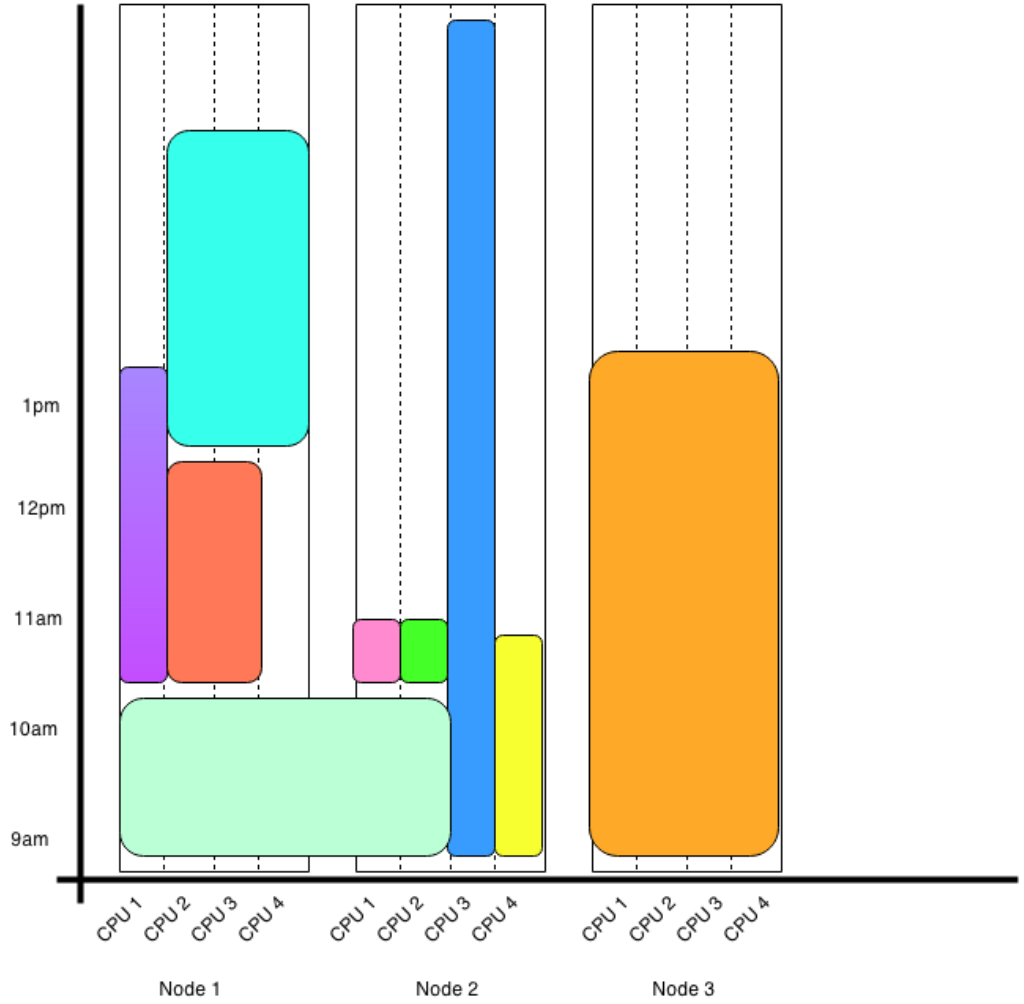
Job Steps

- A user job enters a job queue,
- the scheduler (its strategy) decides on start time and resource allocation of the job.



SLURM Job Scheduler

- **Priority factors**
  - **age**: how long the job has been waiting in the queue,
  - **fair share**: a measure of past usage of the cluster by the user,
  - **size**: the number of CPUs a job requests



Job Classifications

- **Batch Jobs** vs **interactive jobs**
  - batch jobs are queued until execution
  - interactive jobs need immediate resource allocation
- **Parallel** vs. **sequential jobs**
  - a job requires several processing nodes in parallel

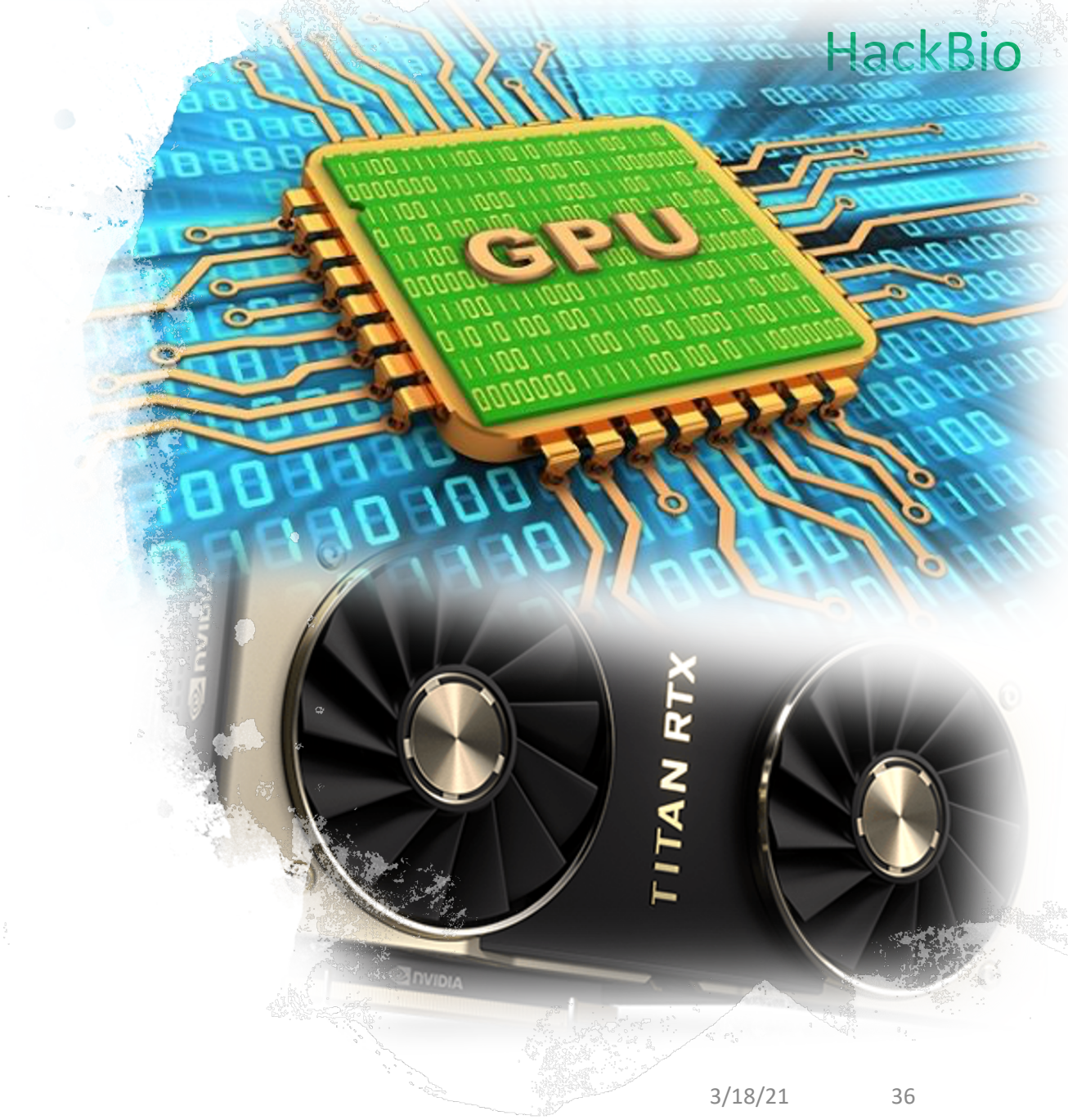
- The majority of HPC installations are used to run batch jobs in space-sharing mode!
  - a job is not influenced by other co-allocated jobs
  - the assigned processors, node memory, caches etc. are exclusively available for a single job.
  - overhead for context switches is minimized
  - important aspects for parallel applications

Accelerators and Many Integrated Cores

- Typical CPUs are not very energy efficient
  - Meant to be general purpose, which requires lots of pieces to the chip
- Accelerators package much more FLOP capabilities with less energy consumption
  - Not exactly general purpose
  - Always requires more work from the programmer
  - Performance improvements and application porting may take significant effort

GPU Accelerators

- A graphics processing unit (GPU) as accelerators
  - NVIDIA GPUs can be used to perform calculations
  - Latest Turing generation offer 130 TFLOPs in a single package

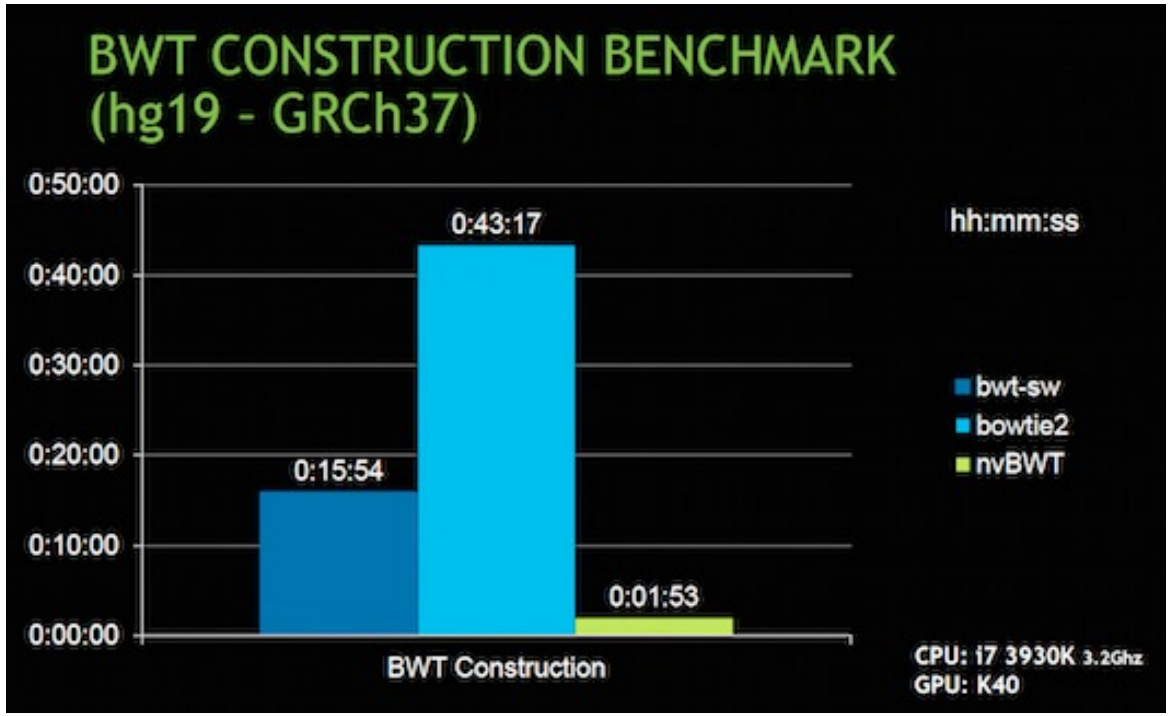


cudasw.sbatch

```
#!/bin/env bash
#SBATCH --job-name=cudasw
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=20
#SBATCH -p longrun
#SBATCH -w keklf-cls01
#SBATCH --gres=gpu:1
#SBATCH --mail-user wgatua@hackbio.org
#Command to run.

module load cudasw/v3.1.1
cudasw -query Queries/all.fasta –db uniprot_sprot.fasta -use_single 1
```

GPU accelerated Bowtie2



- BWT-SW and Bowtie2 compared to nvBWT on human reference genome build *hg19*.
- nvBWT about 8X and 21X faster than BWT-SW and Bowtie2, respectively.

Access to HPC

- Users access the HPC server through ssh connection to their user accounts.
- User account will only be activated after a short induction training session.
- Only you have access to your directory and workspace.

ackBio